UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/771,682	02/04/2004	Gerard M. Col	CNTR.2093	4029

23669 7590 04/10/2006

HUFFMAN LAW GROUP, P.C.
1832 N. CASCADE AVE.
COLORADO SPRINGS, CO 80907-7449

EXAMINER	
MOLL, JESSE R	

ART UNIT	PAPER NUMBER
2181	

DATE MAILED: 04/10/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/771,682	COL, GERARD M.	
	Examiner	Art Unit	
	Jesse R. Moll	2181	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 July 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-54* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-17 and 21-54* is/are rejected.

7)☒ Claim(s) *18-20* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *04 February 2004* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Supervisory

FRITZ FLEMING
PRIMARY EXAMINER
GROUP 2100
AU 2181

4/2/2006

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *13 July 2005*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-54 have been examined.

Acknowledgment of papers filed: oath, specification, drawings, and IDS on February 4, 2004; IDS on September 8, 2004; and IDS on July 13, 2005. The papers filed have been placed on record.

Claim Rejections - 35 USC § 101

2. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

3. Claims 36 and 54 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. Claims 36 and 54 recite the limitation "a computer data signal embodied in a transmission medium". A computer data signal embodied in a transmission medium does not fall under any of the statutory classes. First, a claimed signal is clearly not a "process" under Sec. 101 because it is not a series of steps. The other three Sec. 101 classes of machine, compositions of matter and manufactures "relate to structural entities and can be grouped as `product' claims in order to contrast them with process claims." 1 D. Chisum, Patents Sec. 1.02 (1994). The three product classes have traditionally required physical structure or material.

Examiner suggests (and assumes for the purpose of examination) that the limitation "A computer data signal embodied in a transmission medium" read "A computer program embodied on a computer readable medium". The term computer usable (e.g., readable) medium as defined in Applicant's specification (page 44, lines 1-4) does not include non-tangible signals.

Claim Rejections - 35 USC § 112

1. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

2. Claims 14, 25, 32, and 35 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

3. Claim 14 recites the limitation "said early execution logic resides in a stage of the microprocessor pipeline prior to a stage of the pipeline in which an execution unit for executing a final result of said instruction that precedes the branch instruction resides". It is unclear how two physical objects can be in stages that follow one another. Stage can be interpreted as either a distinct physical part of something, or as a distinct time period. If the term "stage" is interpreted as a physical part, it is unclear how one of the stages is "prior". If the term "stage" is interpreted as a distinct time period, it is unclear how physical objects exist in a period of time. Further, according to Applicant's specification, the architected register file is update via result bus 158 in the W stage

(see paragraph 35, lines 7-8). It is unclear how the architected register file 134 is contained within one stage because it is accessed more than once by the same instruction. For the purpose of examination, Examiner assumes the limitation read "architected status flags are read prior to said early result being generated".

4. Claim 25 recites the limitation "substantially conforming". It is unclear on what substantially conforming means. For the purpose of examination, Examiner assumes the limitation read "conforming".

5. Claim 32 recites the limitation "said early status flags are generated within a stage of the pipeline immediately following a stage of the pipeline including the architected status flags". It is unclear how a physical objects can be in stage that follows a stage in which an event occurs. Stage can be interpreted as either a distinct physical part of something, or as a distinct time period. If the term "stage" is interpreted as a physical part, it is unclear how one of the stages contains generating flags. If the term "stage" is interpreted as a distinct time period, it is unclear how architected status flags exist in a period of time. For the purpose of examination, Examiner assumes the limitation read "said early status flags are generated immediately following architected status flags being read".

6. Claim 35 recites the limitation "wherein a computer program product... causes the apparatus". It is unclear how a computer program on a medium can cause a

processor. Manufacturing a processor is done by a physical machine to print the circuitry. Any code for the processor is merely used by the machine. Further, the term "causes" is unclear because it is not clear if the limitation claims the code creating the processor, or the code causing the processor to do something. For the purpose of examination, Examiner assumes the code causes the processor to operate.

Claims 36 and 54 are rejected because of similar reasons as claim 20. The term "provides" is equally as unclear as "causes". For purpose of examination, Examiner assumes the code provides the apparatus with data.

Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

5. Claims 1-17, 21-24, and 27-54 are rejected under 35 U.S.C. 102(b) as being anticipated by Denman (U.S. Patent No. 5,493,669).

6. Regarding claim 1, Denman discloses an apparatus for correcting a prediction of a branch instruction in a pipeline microprocessor, the prediction predicting whether a branch condition (if the branch should be taken) specified by the branch instruction is satisfied in architected status flags of the microprocessor, the apparatus comprising: early status flags (Results), corresponding to and storing a newer state than the

architected status flags (see col. 4, lines 26-28), said early status flags having a valid indicator (valid bit; see col. 4, lines 63-67; col. 5, lines 1-12); and early branch correction logic (logic which reverses the effects of the branch), for correcting the prediction of the branch instruction (see col. 8, lines 11-15) if said valid indicator indicates said early status flags are valid (operand available in rename buffer; see col. 9, lines 16-19;

Note that in order for the rename buffer to hit, the tag must be in the buffer and that entry must be valid.)

and if said early status flags indicate the prediction is incorrect (see col. 10, lines 11-14).

7. Regarding claim 2, Denman discloses the apparatus of claim 1, further comprising: late branch correction logic (logic for performing operation for flow chart 54), coupled to said early branch correction logic (logic for performing operation for flow chart 54;

Note that the branch correction logic 54 is considered to be both the late branch correction logic and the early branch predictions logic. Further note that inherently, anything is coupled to itself.),

for correcting the prediction of the branch instruction (see col. 10, lines 14-17) if the architected status flags indicate the prediction is incorrect (see col. 9, lines 29-32;

Note that if the data is not in the rename buffer, the architected registers will be used.)

and if said early branch correction logic did not correct the incorrect prediction.

Note that the branch will not be corrected twice. Therefore, the logic will not correct a branch which has already been corrected.

8. Regarding claim 3, Denman discloses the apparatus of claim 2, further comprising: an execution unit (write back logic;

Note that the definition of execution according to WordNet ® 2.0, © 2003 Princeton University is "the process of carrying out an instruction by a computer". Therefore, any unit for carrying out instructions is considered an execution unit. Therefore, the logic for writing results back (completing execution) is considered to be an execution unit.),

coupled to said late branch correction logic, for generating values stored in the architected status flags (see col. 4, lines 26-28).

Note that the definition of generate according to The Free On-line Dictionary of Computing, © 1993-2005 Denis Howe is "To produce something according to an algorithm or program or set of rules, or as a (possibly unintended) side effect of the execution of an algorithm or program." Using this definition, results are generated by the sequencer (see col. 4, lines 29-30) when the results are moved from the rename buffer into the architectural register file.

9. Regarding claim 4, Denman discloses the apparatus of claim 3, wherein said values generated by said execution unit are generated based on valid source operands (see col. 6, lines 36-38 regarding architectural registers always being valid).

Note that since the results are always valid, all source operands must be also valid.

10. Regarding claim 5, Denman discloses the apparatus of claim 2, wherein said early branch correction logic is present in a first stage of the microprocessor pipeline (first half of the branch prediction) and said late branch correction logic is present in a second stage of the microprocessor pipeline (second half of the branch prediction), wherein said second stage is subsequent to said first stage (second half is after the first half).

Note that the definition of the word stage according to WordNet ® 2.0, © 2003 Princeton University is "any distinct time period in a sequence of events". According to this definition, the branch prediction logic is present in both stages since the correction logic must take a non-zero amount of time.

11. Regarding claim 6, Denman discloses the apparatus of claim 1, wherein the architected status flags store program-visible state of the microprocessor (see col. 6, lines 36-38 regarding architectural registers always being valid).

Note that since the operands are always valid, the program will be able to read these values. The rename buffer however, will sometimes contain values which are invalid and therefore not visible by the program.

12. Regarding claim 7, Denman discloses the apparatus of claim 1, wherein the architected status flags are only updated based on results of instructions that the microprocessor has determined are guaranteed to complete (see col. 1, lines 31-34).

Note that instructions are complete when they are written to the architectural register file and therefore are guaranteed to complete.

13. Regarding claim 8, Denman discloses the apparatus of claim 1, wherein the architected status flags are always valid (see col. 6, lines 36-38 regarding architectural registers always being valid).

14. Regarding claim 9, Denman discloses the apparatus of claim 1, wherein said early status flags may be updated based on early results of instructions that the microprocessor has not yet determined are guaranteed to complete (see col. 10, lines 14-17 regarding invalidating entries in the rename buffer).

Note that if a branch is mispredicted, instructions subsequent to the branch corresponding to entries in the rename buffer will not complete.

15. Regarding claim 10, Denman discloses the apparatus of claim 9, wherein an instruction is guaranteed to complete if the microprocessor determines that no instructions preceding said instruction are still capable of generating an exception and all branch instructions preceding said instruction are finally resolved.

Note that in an in-order processor, if no instructions preceding said instructions are capable of generating an exception, the instruction is guaranteed to complete.

16. Regarding claim 11, Denman discloses the apparatus of claim 1, further comprising: logic, coupled to said early status flags (Sequencer; see col. 4, lines 29-30), for invalidating said early status flags (see col. 10, lines 14-17) if an early result of an instruction that precedes the branch instruction and that specifies modification of the status flags is invalid (see col. 10, lines 11-14).

Note that if a branch is mispredicted, the instructions are invalid because they should not have been updated.

17. Regarding claim 12, Denman discloses the apparatus of claim 11, wherein said early result of said instruction that precedes the branch instruction is invalid if said early result is generated based on one or more invalid source operands (see col. 10, lines 12-24).

Note that results of an instruction are valid if all the inputs to that instruction are also valid. If a branch is mispredicted, the results in the rename buffer are invalidated, otherwise the results are valid.

18. Regarding claim 13, Denman discloses the apparatus of claim 11, further comprising: early execution logic (Execution Unit 12; see fig. 1), coupled to said logic,

for generating said early result in response to said instruction that precedes the branch instruction (see col. 3, lines 32-36).

19. Regarding claim 14, Denman discloses the apparatus of claim 13, wherein architected status flags are read prior to said early result being generated (see col. 6, lines 25-45).

20. Regarding claim 15, Denman discloses the apparatus of claim 13, wherein said early result of said instruction that precedes the branch instruction is invalid if said preceding instruction is not included in a set of instructions executable by said early execution logic (see col. 10, lines 12-22).

Note that if an instruction is not executed correctly (it is invalid) it can be considered to not be executable (because it was not executed correctly). One example would be an incorrect branch prediction.

21. Regarding claim 16, Denman discloses the apparatus of claim 15, wherein said set of instructions executable by the early execution logic is smaller than the set of instructions executable by the microprocessor (all instructions which are correctly executed).

22. Regarding claim 17, Denman discloses the apparatus of claim 13, wherein said early execution logic is further configured to generate memory addresses of instruction operands (see col. 4, lines 50-56).

Note that the address of where the operand is to be stored in the rename buffer is generated.

23. Regarding claim 21, Denman discloses the apparatus of claim 1, wherein said early status flags are valid (allocated in the rename buffer; see col. 4, lines 63-67) if said early status flags comprise architected status flag values correctly modified by each uncompleted instruction (instructions not written back) in the pipeline that is an early status flags-modifying instruction and that precedes the branch instruction (see col. 4, lines 26-30).

24. Regarding claim 22, Denman discloses the apparatus of claim 1, wherein said correcting the prediction of the branch instruction comprises flushing the microprocessor pipeline above a stage in which the branch instruction resides and causing the microprocessor to fetch instructions at a corrected instruction address (see col. 8, lines 11-15).

Note that reversing the effects of the branch must include undoing all instructions following the branch instruction. Undoing the branch instructions is considered flushing the instructions.

25. Regarding claim 23, Denman discloses the apparatus of claim 22, wherein said corrected instruction address comprises a next instruction sequentially following the branch instruction if the prediction predicted the branch instruction is taken (see col. 8, lines 15-18).

Note that the correct path of an instruction which was incorrectly predicted as taken is the next instruction.

26. Regarding claim 24, Denman discloses the apparatus of claim 22, wherein said corrected instruction address comprises a branch target address specified by the branch instruction if the prediction predicted the branch instruction is not taken (see col. 8, lines 15-18).

Note that the correct path of an instruction which was incorrectly predicted as not taken is the instruction at the branch target address.

27. Regarding claim 27, Denman discloses the apparatus of claim 1, wherein the prediction of the branch instruction comprises a prediction of whether the branch instruction is taken or not taken (see col. 10, lines 1-11).

28. Regarding claim 28, Denman discloses the apparatus of claim 1, wherein said early status flags indicate the prediction is incorrect if the prediction predicts the branch instruction is taken but the branch condition as indicated in said early status flags is false (see col. 8, lines 8-12).

Note that the branch condition is whether the branch should be taken. Therefore, if the branch is predicted taken, and the processor determines that the branch should not have been taken (condition = false), the prediction is incorrect.

29. Regarding claim 29, Denman discloses the apparatus of claim 1, wherein said early status flags indicate the prediction is incorrect if the prediction predicts the branch instruction is not taken but the branch condition as indicated in said early status flags is true (see col. 8, lines 8-12).

Note that the branch condition is whether the branch should be taken. Therefore, if the branch is predicted not taken, and the processor determines that the branch should have been taken (condition = true), the prediction is incorrect.

30. Regarding claim 30, Denman discloses the apparatus of claim 1, further comprising: a first storage element (Rename Buffer 14; see fig. 1), coupled to said early branch correction logic (see col. 9, lines 20-24), for storing said early status flags (Results); and a second storage element (Valid Bit), coupled to said first storage element, for storing said valid indicator (see col. 4, lines 63-67; col. 5, lines 1-12).

31. Regarding claim 31, Denman discloses the apparatus of claim 1, wherein said early status flags are generated within an address generation stage of the pipeline microprocessor (see col. 4, lines 50-56).

Note that the early status flags are considered to generated within an address generation stage because the address of where the result is to be stored in the rename buffer is calculated.

32. Regarding claim 32, Denman discloses the apparatus of claim 1, wherein said early status flags are generated immediately following architected status flags being read (see col. 6, lines 25-45).

33. Regarding claim 33, Denman discloses the apparatus of claim 1, wherein the pipeline microprocessor is a scalar microprocessor (see col. 1, lines 22-30).

Note that the definition of scalar according to The Free On-line Dictionary of Computing, © 1993-2005 Denis Howe is "Any data type that stores a single value (e.g. a number or Boolean), as opposed to an aggregate data type that has many elements." Using this definition, the processor is considered to be scalar because it is not a vector processor.

34. Regarding claim 34, Denman discloses the apparatus of claim 1, wherein the pipeline microprocessor issues instructions in program order (see col. 5, lines 51-54).

Note that the reorder buffer allocates entries not specified by instructions, but by determining the first empty slot (see col. 7, lines 37-67).

7. Regarding claim 35, Denman discloses the apparatus of claim 1, wherein a computer program product comprising a computer usable medium having computer readable program code causes the apparatus (to operate), wherein said computer program product is for use with a computing device (see col. 2, lines 27-31).

Note that any instructions run on this processor cause the processor to operate.

8. Regarding claim 36, Denman discloses the apparatus of claim 1, wherein a computer data signal embodied in a transmission medium comprising computer-readable program code provides the apparatus (with data) (see col. 2, lines 27-31).

Note that any instructions run on this processor, provide the processor with data.

35. Regarding claim 37, Denman discloses the apparatus of claim 1, wherein the architected status flags store conditions specifiable by conditional program instructions of the microprocessor instruction set (see col. 3, lines 33-36).

Note that according to The American Heritage® Dictionary of the English Language, Fourth Edition the definition of the word conditional is "Imposing, depending on, or containing a condition." It also states that the definition of the word condition is "A mode or state of being". Using these definitions, every instruction in a processor imposes or depends on the state of the processor.

36. Regarding claim 38, Denman discloses a microprocessor having a branch predictor for predicting whether a condition specified by a conditional branch instruction

will be satisfied (see col. 8, lines 1-3), comprising: a storage element (Rename Buffer 14; see fig. 1), for storing status flags of the microprocessor (Results) for use in determining whether the branch condition is satisfied (see col. 9, lines 20-28), wherein said status flags are invalid if an instruction preceding the branch instruction specifies an operation that is not executable by early execution logic of the microprocessor (see col. 10, lines 12-22);

Note that if an instruction is not executed correctly (it is invalid) it can be considered to not be executable (because it was not executed correctly). One example would be an incorrect branch prediction before said branch instruction.

And branch correction logic (logic which reverses the effects of the branch), coupled to said storage element (see col. 9, lines 20-24), for correcting the prediction (see col. 8, lines 11-15) if said status flags are valid (operand available in rename buffer; see col. 9, lines 16-19)

Note that in order for the rename buffer to hit, the tag must be in the buffer and that entry must be valid.

And said status flags indicate the branch predictor mispredicted whether the condition specified by the conditional branch instruction is satisfied (see col. 10, lines 11-14), and for generating an indication of whether said branch correction logic corrected the branch instruction for use by subsequent final branch correction logic (see col. 10, lines 14-17).

Note that the branch correction logic must know that the branch has been mispredicted in order to correct it. Whatever signifies that the branch has been

mispredicted is considered to be the indication. Further note that the branch correction logic and the final branch correction logic are considered to be the same thing.

37. Regarding claim 39, Denman discloses the microprocessor of claim 38, wherein said early execution logic is in a same stage of the microprocessor pipeline as said storage element.

Note that according to The American Heritage® Dictionary of the English Language, Fourth Edition, *the definition of the word stage is "An element or a group of elements in a complex arrangement of parts". Examiner considers the entire processor a stage (a stage to execute instructions).*

38. Regarding claim 40, Denman discloses the microprocessor of claim 38, wherein said status flags are invalid if modified based on results of an instruction preceding the branch instruction generated using invalid input operands (see col. 10, lines 12-24).

Note that results of an instruction are valid if all the inputs to that instruction are also valid. If a branch is mispredicted, the results in the rename buffer are invalidated, otherwise the results are valid.

39. Regarding claim 41, Denman discloses the microprocessor of claim 38, wherein said subsequent final branch correction logic uses second status flags that are always valid for use in determining whether the branch condition is satisfied (see col. 9, lines 29-33).

Note that if values do not exist in the rename buffer, then results will be used from the architectural buffer.

40. Regarding claim 42, Denman discloses the microprocessor of claim 38, further comprising: an early register file (rename buffer; see fig. 1), coupled to provide operands to said early execution logic for generation of a result of said preceding instruction (see col. 3, lines 33-36), wherein said status flags are modified in response to said result, wherein said status flags are invalid if said early register file provides an operand to said early execution logic that is invalid (see col. 10, lines 11-14).

Note that if a branch is mispredicted and it causes the input operand of the instruction to become invalid the status flags would be invalidated.

41. Regarding claim 43, Denman discloses the microprocessor of claim 42, wherein said early register file is further coupled to receive results generated by said early execution logic (see col. 3, lines 33-36).

42. Regarding claim 44, Denman discloses the microprocessor of claim 38, wherein if at least one input operand provided to said preceding instruction is invalid, said status flags are invalid (see col. 10, lines 11-14).

Note that the branch correction logic invalidates all entries in the rename buffer that are affected by a mispredicted branch.

43. Claim 45 recites equivalent limitations as claim 31 and is therefore rejected under the same grounds.

44. Claim 46 recites equivalent limitations as claim 33 and is therefore rejected under the same grounds.

45. Claim 47 recites equivalent limitations as claim 34 and is therefore rejected under the same grounds.

46. Regarding claim 48, Denman discloses a method for correcting a prediction of a conditional branch instruction early in a microprocessor pipeline, the method comprising: determining whether early status flags are valid (operand available in rename buffer; see col. 9, lines 16-19);

Note that in order for the rename buffer to hit, the tag must be in the buffer and that entry must be valid.

Determining whether the prediction is incorrect (see col. 10, lines 11-14) based on whether the early status flags satisfy a branch condition specified by the conditional branch instruction; and correcting the prediction of the conditional branch instruction if the early status flags are valid and the prediction is incorrect (see col. 8, lines 11-15) based on whether the early status flags satisfy a branch condition specified by the conditional branch instruction (see col. 9, lines 20-24).

Note that it will always correct the branch prediction if the prediction is incorrect. Therefore, it will still correct the prediction if the status flags are valid.

47. Regarding claim 49, Denman discloses the method of claim 48, further comprising: generating the early status flags, prior to said determining whether early status flags are valid (see col. 10, lines 14-18).

Note that if entries are invalidated, the entries must have been generated before they were determined to be invalid or valid.

48. Regarding claim 50, Denman discloses the method of claim 49, wherein said generating the early status flags comprises: generating the early status flags in response to generating an early result of an instruction preceding the conditional branch instruction (see col. 4, lines 26-28).

49. Regarding claim 51, Denman discloses the method of claim 50, wherein the preceding instruction immediately precedes the conditional branch instruction by a single pipeline stage of the microprocessor.

Note that every instruction result is forwarded to the rename buffer. Therefore, the instruction immediately preceding the conditional branch instruction by a single pipeline stage will also send its result to the rename buffer.

50. Claim 52 recites equivalent limitations as claim 31 and is therefore rejected under the same grounds.

51. Regarding claim 53, Denman discloses the method of claim 48, wherein the early status flags comprise non-architected status flags of the microprocessor (see col. 3, lines 19-21).

Note that the rename buffer is not the architectural register file.

52. Note that claim 54 recites equivalent limitations as claim 1 and is therefore rejected under the same grounds. Note that the code for providing the processor data is all instructions executed on the processor (see above regarding claim 36).

Claim Rejections - 35 USC § 103

53. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

54. Claims 25 is rejected under 35 U.S.C. 103(a) as being unpatentable over Denman in view of Intel3 (IA-32 Intel Architecture Software Developer's Manual Volume 3: System Programming Guide) herein referred to as Intel3.

Denman discloses the apparatus of claim 1.

Denman does not explicitly disclose the architected status flags comprise status flags stored in a register conforming to an x86 architecture EFLAGS register.

Intel3 teaches said architected status flags comprise one or more x86 architecture EFLAGS register status flags (see pages 2-7, 2-8 and 2-9; section 2.3). It would have been obvious at the time of the invention for one of ordinary skill in the art to have modified the invention of Denman by including one or more x86 architecture EFLAGS register status flags in said architected status flags, as taught by Intel3, in order to control I/O, maskable hardware interrupts, debugging, task switching, and the virtual-8086 mode (see last 3 lines of page 2-7).

55. Claims 26 is rejected under 35 U.S.C. 103(a) as being unpatentable over Denman in view of Intel2 (Intel Architecture Software Developer's Manual Volume 2: Instruction Set Reference) herein referred to as Intel2.

Denman discloses the apparatus of claim 1.

Denman does not explicitly disclose the branch instruction comprises an x86 architecture conditional branch instruction.

Intel2 teaches the branch instruction comprises an x86 architecture conditional branch instruction (pages 3-354, 3-355, 3-336 and 3-357).

It would have been obvious at the time of the invention for one of ordinary skill in the art to have modified the invention of Denman by using x86 architecture conditional branch instruction, as taught by Intel2, in order to conditionally branch. It is well known

in the art that any significant program contains conditional branches. Conditional branches allow for multiple control paths in a program.

Allowable Subject Matter

56. Claims 18-20 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

57. The following is a statement of reasons for the indication of allowable subject matter:

58. Claim 18 recites the limitation "said logic copies the architected status flags to said early status flags and validates said early status flags if the microprocessor pipeline is flushed." The prior art does not teach or fairly suggest updating temporary status flags (as recited in claim 1) by copying architected status flags when the microprocessor pipeline is flushed.

59. Claim 19 recites the limitation "said logic copies the architected status flags to said early status flags and validates said early status flags if no instructions in the pipeline between a first and second stage of the pipeline are instructions that require modification of the status flags." The prior art does not teach or fairly suggest updating

temporary status flags (as recited in claim 1) by copying architected status flags when all status flags modifying instructions present in the pipeline between two stages.

60. Claim 20 is indicated as allowable subject matter because of its dependence on claim 19, which is also indicated as allowable subject matter.

Conclusion

61. The following is text cited from 37 CFR 1.11(c): In amending in reply to a rejection of claims in an application or patent under reexamination, the applicant or patent owner must clearly point out the patentable novelty which he or she thinks the claims present in view of the state of the art disclosed by the references cited or the objections made. The applicant or patent owner must also show how the amendments avoid such references or objections.

62. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jesse R. Moll whose telephone number is (571)272-2703. The examiner can normally be reached on M-F 8:00 am - 4:30 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Fritz M. Fleming can be reached on 571-272-4145. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

***

Fritz M. Fleming
FRITZ FLEMING
Supervisory PRIMARY EXAMINER
GROUP 2100
AU 2181
4/2/2006